Double Lasso

knitr::opts\_chunk$set(echo = TRUE)  
#+ setup, error=TRUE

## Data Preparation

# Brian Kang  
# Must use 64-bit version of R  
#knitr::opts\_chunk$set(error=TRUE)  
#rm(list = ls())  
  
setwd("H:/Honors Research")  
#chooseCRANmirror(graphics=FALSE, ind=1)  
#install.packages(c("hdm","dplyr","stringr","glmnet","gdata","fastDummies"))  
library(hdm)  
library(dplyr)

##   
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':  
##   
## filter, lag

## The following objects are masked from 'package:base':  
##   
## intersect, setdiff, setequal, union

library(stringr)  
library(glmnet)

## Loading required package: Matrix

## Loaded glmnet 3.0-2

library(gdata)

## gdata: Unable to locate valid perl interpreter  
## gdata:   
## gdata: read.xls() will be unable to read Excel XLS and XLSX files  
## gdata: unless the 'perl=' argument is used to specify the location of a  
## gdata: valid perl intrpreter.  
## gdata:   
## gdata: (To avoid display of this message in the future, please ensure  
## gdata: perl is installed and available on the executable search path.)

## gdata: Unable to load perl libaries needed by read.xls()  
## gdata: to support 'XLX' (Excel 97-2004) files.

##

## gdata: Unable to load perl libaries needed by read.xls()  
## gdata: to support 'XLSX' (Excel 2007+) files.

##

## gdata: Run the function 'installXLSXsupport()'  
## gdata: to automatically download and install the perl  
## gdata: libaries needed to support Excel XLS and XLSX formats.

##   
## Attaching package: 'gdata'

## The following objects are masked from 'package:dplyr':  
##   
## combine, first, last

## The following object is masked from 'package:stats':  
##   
## nobs

## The following object is masked from 'package:utils':  
##   
## object.size

## The following object is masked from 'package:base':  
##   
## startsWith

library(fastDummies)  
library(lmtest)

## Loading required package: zoo

##   
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':  
##   
## as.Date, as.Date.numeric

library(car)

## Loading required package: carData

##   
## Attaching package: 'car'

## The following object is masked from 'package:dplyr':  
##   
## recode

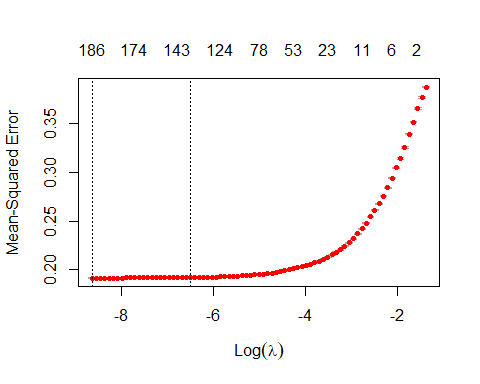
#dat <- read.table("e20185wa0002000.txt", sep = ",") # 5-year ACS survey Washington data  
  
# Import dataset ---------------------------------------------------------------------------------------------------  
temp <- read.csv("psam\_pusb.csv", header = T, nrows = 1)  
# columns that don't import as factors using default settings  
fnf <- c("DIVISION","PUMA","REGION","ST","ADJINC","CIT","COW","DDRS","DEAR",  
 "DEYE","DOUT","DPHY","DRAT","DRATX","DREM","ENG","FER","GCL","GCM","GCR",  
 "HINS1","HINS2","HINS3","HINS4","HINS5","HINS6","HINS7","JWTR","LANX",  
 "MAR","MARHD","MARHM","MARHT","MARHW","MIG","MIL","MLPA","MLPB","MLPCD",  
 "MLPE","MLPFG","MLPH","MLPI","MLPJ","MLPK","NWAB","NWAV","NWLA","NWLK",  
 "NWRE","RELP","SCH","SCHG","SCHL","SEX","WKL","WKW","WRK","ANC","ANC1P",  
 "ANC2P","DECADE","DIS","DRIVESP","ESP","ESR","FOD1P","FOD2P","HICOV",  
 "HISP","INDP","JWAP","JWDP","LANP","MIGPUMA","MIGSP","MSP","NATIVITY",  
 "NOP","OC","OCCP","PAOC","POBP","POWPUMA","POWSP","PRIVCOV","PUBCOV",  
 "QTRBIR","RAC1P","RAC2P","RAC3P","RACAIAN","RACASN","RACBLK","RACNH",  
 "RACNUM","RACPI","RACSOR","RACWHT","RC","SCIENGP","SCIENGRLP","SFN",  
 "SFR","VPS","WAOB")  
# columns that do import as factors using default setting  
fif <- c("RT","SERIALNO","NAICSP","SOCP")  
# all columns that are factors  
fcf <- append(fnf,fif)  
fcf <- append(fcf,names(temp[,c(131:286)]))  
# vector of classes of data columns  
colclass <- ifelse(colnames(temp) %in% fcf, 'factor', 'numeric')  
  
temp1 <- read.csv("psam\_pusa.csv", header = T, colClasses = colclass) # U.S. PUMS data  
temp2 <- read.csv("psam\_pusb.csv", header = T, colClasses = colclass)  
dat <- rbind(temp1,temp2)  
dat <- dat[,-c(1,2)] # drop unecessary IDs  
dat <- dat[,-c(129:284)] # drop unnecessary flag vars  
  
# agep: age of person  
# SUM (intp, oip, pap, retp, semp, ssip, ssp,  
# wagp) =   
# pernp, for total income  
  
# US Census: "Income used to calculate poverty status includes PERNP (earnings) and PINCP (income)"  
  
# Calculate Income-Poverty ratio ----------------------------------------------------------------------------------  
# (POVPIP only shows NA or <0.5 or >=0.5 so calculate actual ratio)  
PovertyThreshold <- rep(NA, nrow(dat))  
getThreshold <- function(threshold) { # values from CPS 2018  
 for (i in 1:nrow(dat)) {  
 if (dat$AGEP[i] < 18) { # under 18 yrs  
 threshold[i] <- NA  
 } else if (dat$SPORDER[i]==1 & dat$AGEP[i] < 65) { # individual  
 threshold[i] <- 13064  
 } else if (dat$SPORDER[i]==1 & dat$AGEP[i] >= 65) {  
 threshold[i] <- 12043  
 # AGEP doesn't have NA values  
 } else if (dat$SPORDER[i]==2 & dat$AGEP[i] < 65 & dat$OC[i]==0) { # two people  
 threshold[i] <- 16815  
 } else if (dat$SPORDER[i]==2 & dat$AGEP[i] < 65 & dat$OC[i]==1) {  
 threshold[i] <- 17308  
 } else if (dat$SPORDER[i]==2 & dat$AGEP[i] >= 65 & dat$OC[i]==0) {  
 threshold[i] <- 15178  
 } else if (dat$SPORDER[i]==2 & dat$AGEP[i] >= 65 & dat$OC[i]==1) {  
 threshold[i] <- 17242  
 } else if (dat$SPORDER[i]==2) { # OC is NA value  
 threshold[i] <- 16247  
 } else if (dat$SPORDER[i]==3 & dat$OC[i]==0) { # three people  
 threshold[i] <- 19642  
 } else if (dat$SPORDER[i]==3 & dat$OC[i]==1) {  
 threshold[i] <- (20212+20231)/2  
 } else if (dat$SPORDER[i]==3) { # OC is NA value  
 threshold[i] <- 19985  
 } else if (dat$SPORDER[i]==4 & dat$OC[i]==0) { # four people  
 threshold[i] <- 25900  
 } else if (dat$SPORDER[i]==4 & dat$OC[i]==1) {  
 threshold[i] <- (26324+25465+25554)/3  
 } else if (dat$SPORDER[i]==4) { # OC is NA value  
 threshold[i] <- 25701  
 } else if (dat$SPORDER[i]==5 & dat$OC[i]==0) { # five people  
 threshold[i] <- 31234  
 } else if (dat$SPORDER[i]==5 & dat$OC[i]==1) {  
 threshold[i] <- (31689+30718+29967+29509)/4  
 } else if (dat$SPORDER[i]==5) { # OC is NA value  
 threshold[i] <- 30459  
 } else if (dat$SPORDER[i]==6 & dat$OC[i]==0) { # six people  
 threshold[i] <- 35925  
 } else if (dat$SPORDER[i]==6 & dat$OC[i]==1) {  
 threshold[i] <- (36068+35324+34612+33553+32925)/5  
 } else if (dat$SPORDER[i]==6) { # OC is NA value  
 threshold[i] <- 34533  
 } else if (dat$SPORDER[i]==7 & dat$OC[i]==0) { # seven people  
 threshold[i] <- 41336  
 } else if (dat$SPORDER[i]==7 & dat$OC[i]==1) {  
 threshold[i] <- (4159+40705+40085+38929+37581+36102)/6  
 } else if (dat$SPORDER[i]==7) { # OC is NA value  
 threshold[i] <- 39194  
 } else if (dat$SPORDER[i]==8 & dat$OC[i]==0) { # eight people  
 threshold[i] <- 46231  
 } else if (dat$SPORDER[i]==8 & dat$OC[i]==1) {  
 threshold[i] <- (46640+45800+45064+44021+42696+41317+40967)/7  
 } else if (dat$SPORDER[i]==8) { # OC is NA value  
 threshold[i] <- 43602  
 } else if (dat$SPORDER[i]>=9 & dat$OC[i]==0) { # nine or more people  
 threshold[i] <- 55613  
 } else if (dat$SPORDER[i]>=9 & dat$OC[i]==1) {  
 threshold[i] <- (55883+55140+54516+53491+52082+50807+50491+48546)/8  
 } else if (dat$SPORDER[i]>=9) { # OC is NA value  
 threshold[i] <- 51393  
 } else {  
 threshold[i] <- NA  
 }  
 } # individually assign poverty threshold  
 return(threshold)  
}  
PovertyThreshold <- getThreshold(PovertyThreshold)  
dat$IncomePovertyRatio <- (dat$PERNP + dat$PINCP)/PovertyThreshold  
  
#summary(dat$IncomePovertyRatio) # there are negative values, will cause problems for log()  
dat$IncomePovertyRatio <- dat$IncomePovertyRatio + 1 + abs(min(dat$IncomePovertyRatio, na.rm=na.omit)) # ensure all values are positive  
  
  
# function for catching error  
myTryCatch <- function(expr) {  
 warn <- err <- NULL  
 value <- withCallingHandlers(  
 tryCatch(expr, error=function(e) {  
 err <<- e  
 NULL  
 }), warning=function(w) {  
 warn <<- w  
 invokeRestart("muffleWarning")  
 })  
 list(error=err)  
}

## Endogenous Double ML

## Initial Model 2: Double ML (Double LASSO) ----------------------------------------------------------------  
# Based on MODEL 3 from "main.R" (econ483 code)  
temp <- dat  
dat <- temp # recover original dataset  
  
# DATA CLEANING FOR DOUBLE LASSO  
nums <- unlist(lapply(dat, is.factor))  
dattemp <- dat[,nums] # all the factors of dataset  
# delete variables have >51 factor levels  
get1 <- names(which(sapply(dattemp, function(x) length(unique(x))>51)))  
delete <- which(names(dat) %in% get1[-16])  
dat <- dat[,-c(delete)]  
dat <- dat[,-c(9,43:52,82,90,108,95,100)] # delete CITWP,MIL~MLPK,HISP,POVPIP,SFR,RAC1P,RACNUM  
dat <- dat[,-c(1,3,76,77)] # delete DIVISION,REGION,PERNP,PINCP  
# delete NA rows for Income-Poverty ratio  
dat <- dat[!is.na(dat$IncomePovertyRatio),]  
#which(colnames(dat)=="JWRIP") # likely unrelated to overall model, redundant  
#which(colnames(dat)=="YOEP") # too many NAs  
dat <- dat[,-c(which(colnames(dat)=="JWRIP"),which(colnames(dat)=="YOEP"))]  
# delete NA rows for JWMNP (travel time to work)  
dat <- dat[!is.na(dat$JWMNP),]  
# delete NA rows for MARHYP: for factor <2 error  
dat <- dat[!is.na(dat$MARHYP),]  
# delete NA rows for WKHP (usual hours worked per week)  
dat <- dat[!is.na(dat$WKHP),]  
#colSums(is.na(dat))  
  
  
# delete/add vars that should be deleted/added (from correlation/causal inference)  
dat <- dat[, -which(names(dat) %in% c("INTP","OIP","PAP","RETP","SEMP","SSIP","SSP","WAGP"))] # drop values related to Income  
dat <- dat[, -which(names(dat) %in% c("SCH","SCHG"))] # drop weird educ vars  
dat <- dat[, -which(names(dat) %in% c("ANC"))] # drop ancestry  
dat <- droplevels(dat)  
#str(dat)  
dat <- dat[, -which(names(dat) %in% c("ESR","ESP"))] # drop meaningless labor vars  
dat <- dat[, -which(names(dat) %in% c("HICOV","PRIVCOV","PUBCOV"))] # drop redundant insurance vars  
dat <- dat[, -which(names(dat) %in% c("OC","RC"))] # drop child vars. Lack data  
dat <- dat[, -which(names(dat) %in% c("SFN"))] # drop "subfamily number"  
dat <- dat[, -which(names(dat) %in% c("WRK"))] # drop "worked last week" (we don't know when is "last week")  
  
dat <- cbind(dat, model.matrix(~(AGEP:HINS3), dat)[,-1]) # age\*medicare  
dat <- cbind(dat, model.matrix(~(SCIENGP:SCHL), dat)[,-1]) # stem degree\*attained degree  
dat <- cbind(dat, model.matrix(~(SCIENGRLP:SCHL), dat)[,-1]) # stem related degree\*attained degree  
dat$VETERAN <- ifelse((dat$DRATX %in% c("1","2")), 1, 0) # veteran or not  
dat <- dat[, -which(names(dat) %in% c("DRATX","VPS","DRAT"))] # drop veteran related vars  
dat <- cbind(dat, model.matrix(~(AGEP:VETERAN), dat)[,2]) # age\*veteran or not  
names(dat)[ncol(dat)] <- "AGEP\_VETERAN"  
dat <- cbind(dat, model.matrix(~(AGEP:GCL), dat)[,-1]) # age\*grandparent living with grandchild  
dat <- cbind(dat, model.matrix(~(AGEP:GCR), dat)[,-1]) # age\*grandparent responsible grandchild  
  
  
# MUST INCLUDE THIS LINE OF CODE MUST INCLUDE THIS LINE OF CODE MUST INCLUDE THIS LINE OF CODE  
# logical dummy for ST==POWSP: most are FALSE  
dat$SameResidenceWorkplace <- (as.numeric(dat$ST)==as.numeric(dat$POWSP))  
dat <- dat[,-which(names(dat) %in% c("ST","POWSP"))] # delete ST,POWSP  
#colSums(is.na(dat))  
# get which variables have <2 factor levels  
get2 <- which(sapply(dat, function(x) length(unique(x)))<2)  
dat <- dat[,-get2]  
names(dat) <- str\_replace(names(dat), ":", "\_") # reformat interaction term names  
  
# STEP 1: FIRST LASSO: LOG(IncPovRatio) on ALL POTENTIAL VARIATES (i.e. y on controls)  
varnames <- paste(c(names(dat[,-c(which(names(dat) %in%   
 c("IncomePovertyRatio","SameResidenceWorkplace",  
 "JWMNP","JWTR")))])), collapse = "+")  
# throw in everything and see what happens with this LASSO  
formula <- paste(c("log(IncomePovertyRatio)",varnames), collapse = "~")  
#cat("LASSO Formula for y ~ control variables\n")  
#formula  
  
# Split data into train and test for K-fold CV  
set.seed(497)  
train <- sample(1:nrow(dat), nrow(dat)\*0.8) # 80% for training  
  
# get which variables have <2 factor levels AFTER SUBSETTING  
get <- which(sapply(dat[train,], function(x) length(unique(x))<2))  
  
# get train and test datasets  
# takeout intercept  
xtrain <- model.matrix(as.formula(formula), data = dat[train,])[,-1]  
#xtest <- model.matrix(as.formula(formula), data = dat[-train,])[,-1]  
ytrain <- log(dat[train,]$IncomePovertyRatio)  
  
# cross validation then fit LASSO  
cv.lasso.1 <- cv.glmnet(xtrain, ytrain, alpha = 1) # 1 for lasso  
#cat("CV on LASSO and get min tuning parameter\n")  
#cv.lasso.1[c(8,9)]  
cv.lambda.1 <- cv.lasso.1$lambda.min # get smallest tuning parameter  
#cat("\nLambda chosen from K-fold CV\n")  
cv.lambda.1

## [1] 0.0001775132

#png(filename="lassoCV.png") # save plot  
plot(cv.lasso.1)



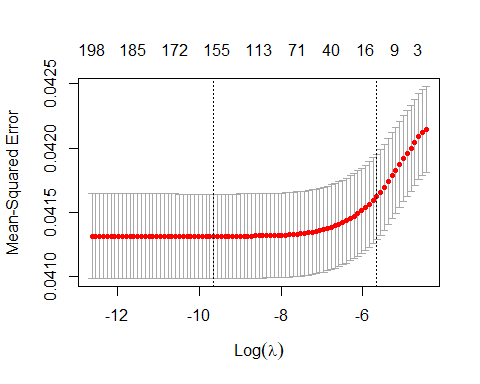
# run lasso and get the necessary focal variables  
dlasso.1 <- rlasso(formula, data = dat[train,],   
 lambda.start = cv.lambda.1, post = F)  
#summary(lasso.2, all = F)  
control <- which(coef(dlasso.1)[-1]!=0)  
#cat("\nHow many variables to keep\n")  
length(control)

## [1] 157

# STEP 2: SECOND LASSO: Core vars on ALL POTENTIAL VARIATES (i.e. focal on controls)  
formula2.1 <- paste(c("SameResidenceWorkplace",varnames), collapse = "~")  
# k-fold cv  
xtrain <- model.matrix(as.formula(formula2.1), data = dat[train,])[,-1]  
ytrain <- dat[train,]$SameResidenceWorkplace  
cv.lasso.2.1 <- cv.glmnet(xtrain, ytrain, alpha = 1) # 1 for lasso  
cv.lambda.2.1 <- cv.lasso.2.1$lambda.min # get smallest tuning parameter  
cv.lambda.2.1

## [1] 6.405355e-05

plot(cv.lasso.2.1)



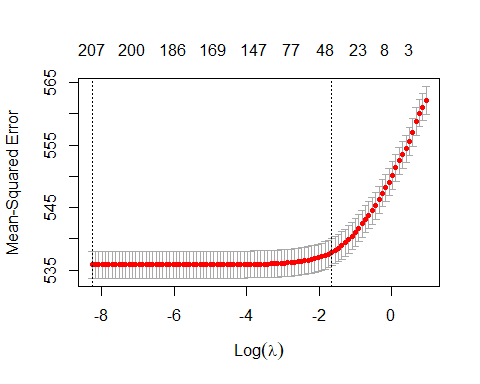
# lasso  
dlasso.2.1 <- rlasso(formula2.1, data = dat[train,],   
 lambda.start = cv.lambda.2.1, post = F)  
#summary(dlasso.2.1, all = F)  
focal1 <- which(coef(dlasso.2.1)[-1]!=0)  
length(focal1)

## [1] 74

# travel time  
formula2.2 <- paste(c("JWMNP",varnames), collapse = "~")  
# k-fold cv  
xtrain <- model.matrix(as.formula(formula2.2), data = dat[train,])[,-1]  
ytrain <- dat[train,]$JWMNP  
cv.lasso.2.2 <- cv.glmnet(xtrain, ytrain, alpha = 1) # 1 for lasso  
cv.lambda.2.2 <- cv.lasso.2.2$lambda.min # get smallest tuning parameter  
cv.lambda.2.2

## [1] 0.0002590354

plot(cv.lasso.2.2)



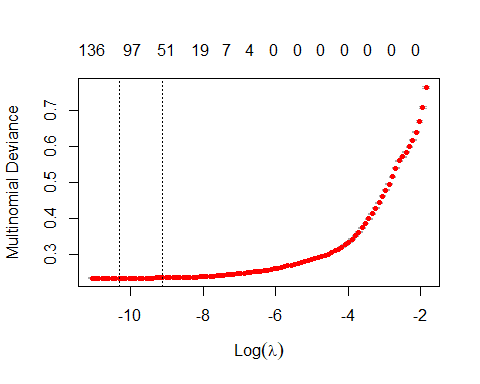
# lasso  
dlasso.2.2 <- rlasso(formula2.2, data = dat[train,],   
 lambda.start = cv.lambda.2.2, post = F)  
#summary(dlasso.2.2, all = F)  
focal2 <- which(coef(dlasso.2.2)[-1]!=0)  
length(focal2)

## [1] 81

formula2.4 <- paste(c("JWTR",varnames), collapse = "~")  
# k-fold cv  
xtrain <- model.matrix(as.formula(formula2.4), data = dat[train,])[,-1]  
ytrain <- drop.levels(dat[train,]$JWTR) # factor level "11" has 0 observations, use drop.levels()  
cv.lasso.2.4 <- cv.glmnet(xtrain, ytrain, alpha = 1, family = "multinomial", nfolds = 3) # 1 for lasso  
cv.lambda.2.4 <- cv.lasso.2.4$lambda.min # get smallest tuning parameter  
cv.lambda.2.4

## [1] 3.31276e-05

plot(cv.lasso.2.4)



# lasso  
tempdat2 <- fastDummies::dummy\_cols(dat)  
focal4 <- c()  
for (ii in 1:11) {  
 if (ii<10) {  
 formula2.4.n <- paste(c(paste("JWTR\_0",ii,sep=""),varnames), collapse = "~")  
 dlasso.2.4.n <- rlasso(formula2.4.n, data = tempdat2[train,],   
 lambda.start = cv.lambda.2.4, post = F)  
 focal4.n <- which(coef(dlasso.2.4.n)[-1]!=0)  
 focal4 <- unique(c(focal4, names(focal4.n)))  
 } else if (ii==10) {  
 formula2.4.n <- paste(c("JWTR\_10",varnames), collapse = "~")  
 dlasso.2.4.n <- rlasso(formula2.4.n, data = tempdat2[train,],   
 lambda.start = cv.lambda.2.4, post = F)  
 focal4.n <- which(coef(dlasso.2.4.n)[-1]!=0)  
 focal4 <- unique(c(focal4, names(focal4.n)))  
 } else if (ii==11) {  
 formula2.4.n <- paste(c("JWTR\_12",varnames), collapse = "~")  
 dlasso.2.4.n <- rlasso(formula2.4.n, data = tempdat2[train,],   
 lambda.start = cv.lambda.2.4, post = F)  
 focal4.n <- which(coef(dlasso.2.4.n)[-1]!=0)  
 focal4 <- unique(c(focal4, names(focal4.n)))  
 }  
}  
length(focal4)

## [1] 213

# STEP 3: Take union of all remainder potential variates  
union <- c(names(control), names(focal1), names(focal2), focal4)  
if (any(duplicated(union))==T) {  
 union <- unique(union)  
}  
  
#cat("\nTotal number of feature variables kept from Double Lasso\n")  
length(union)

## [1] 219

# STEP 4: do OLS of y on focals and kept potential variates  
unionf <- paste(c("SameResidenceWorkplace\*JWMNP+JWTR\*JWMNP",union), collapse = "+")  
formula <- paste(c("log(IncomePovertyRatio)", unionf), collapse = "~")  
  
# name all extra variables created from doing LASSO  
dattemp <- dat  
#olsDLasso1<- lm(formula, data = dattemp)  
# run this code if rlasso created extra dummy vars  
for (i in 1:500) { # look at formula and count how many new vars need to be made  
 error <- myTryCatch(olsDLasso1<- lm(formula, data = dattemp)) # CAUTION  
 newvars <- substr(error[[1]], 45, str\_length(error[[1]])-12)  
 existingvars <- names(dattemp)[which(str\_detect(newvars, names(dattemp)))]  
 existingchars <- sub(existingvars, "", newvars)  
 dattemp[,newvars] <- dattemp[,which(names(dattemp)==existingvars)]==existingchars  
}

## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing

## Warning in sub(existingvars, "", newvars): argument 'pattern' has length > 1 and  
## only the first element will be used

## Warning in names(dattemp) == existingvars: longer object length is not a  
## multiple of shorter object length

## Warning in stri\_length(string): argument is not an atomic vector; coercing

## Warning in sub(existingvars, "", newvars): argument 'pattern' has length > 1 and  
## only the first element will be used

## Warning in stri\_length(string): argument is not an atomic vector; coercing

## Warning in sub(existingvars, "", newvars): argument 'pattern' has length > 1 and  
## only the first element will be used

## Warning in names(dattemp) == existingvars: longer object length is not a  
## multiple of shorter object length

## Warning in stri\_length(string): argument is not an atomic vector; coercing

## Warning in sub(existingvars, "", newvars): argument 'pattern' has length > 1 and  
## only the first element will be used

## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing

## Warning in sub(existingvars, "", newvars): argument 'pattern' has length > 1 and  
## only the first element will be used

## Warning in names(dattemp) == existingvars: longer object length is not a  
## multiple of shorter object length

## Warning in stri\_length(string): argument is not an atomic vector; coercing

## Warning in sub(existingvars, "", newvars): argument 'pattern' has length > 1 and  
## only the first element will be used

## Warning in stri\_length(string): argument is not an atomic vector; coercing

## Warning in sub(existingvars, "", newvars): argument 'pattern' has length > 1 and  
## only the first element will be used

## Warning in names(dattemp) == existingvars: longer object length is not a  
## multiple of shorter object length

## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing

## Warning in sub(existingvars, "", newvars): argument 'pattern' has length > 1 and  
## only the first element will be used

## Warning in names(dattemp) == existingvars: longer object length is not a  
## multiple of shorter object length

## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing

## Warning in sub(existingvars, "", newvars): argument 'pattern' has length > 1 and  
## only the first element will be used

## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing

## Warning in sub(existingvars, "", newvars): argument 'pattern' has length > 1 and  
## only the first element will be used

## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing  
  
## Warning in stri\_length(string): argument is not an atomic vector; coercing

## Error in sub(existingvars, "", newvars): invalid 'pattern' argument

# start with declairing the new vars  
which(colSums(is.na(dattemp))==nrow(dattemp))

## SCIENGRLP1 SCIENGRLP2   
## 238 239

dattemp$MARHT3 <- dattemp$MARHT == "3"  
dattemp$MARHD2 <- dattemp$MARHD == "2"  
dattemp$MARHD8 <- dattemp$MARHD == "8"  
dattemp$MARHT2 <- dattemp$MARHT == "2"  
  
  
which(lapply(dattemp, class)=="matrix")

## named integer(0)

dattemp$SCIENGRLP1 <- dattemp$SCIENGRLP == "1"  
dattemp$SCIENGRLP2 <- dattemp$SCIENGRLP == "2"  
  
# multicolinearity: get which variables have <2 unique values  
multicol <- names(which(sapply(dattemp[train,], function(x) length(unique(x))<2)))  
# manually delete some of the rest (NA values in summary of lm, multicollinearity)  
multicol <- c(multicol,  
 "MSP3","MSP4","MSP5","ENG1","SCHL21","DRIVESP6",  
 "NATIVITY2","SCHL18","DECADE6","WAOB4")  
union <- union[-which(union %in% multicol)] # delete them from formula  
aliased <- which(summary(lm(formula, data = dattemp[train,]))$aliased)  
union <- union[-which(union %in% names(aliased))]  
  
# rewrite formula for OLS  
unionf <- paste(c("SameResidenceWorkplace\*JWMNP+JWTR\*JWMNP",union), collapse = "+")  
formula <- paste(c("log(IncomePovertyRatio)", unionf), collapse = "~")  
  
# Training OLS regression post LASSO  
olsDLasso1 <- lm(formula, data = dattemp[train,])  
DMLresult <- summary(olsDLasso1)  
#cat("\nPost-Double LASSO OLS Result\n")  
DMLresult

##   
## Call:  
## lm(formula = formula, data = dattemp[train, ])  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -3.0233 -0.2576 -0.0282 0.2212 3.5316   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 1.918e+00 1.461e-01 13.124 < 2e-16 \*\*\*  
## SameResidenceWorkplaceTRUE -8.169e-02 4.054e-03 -20.149 < 2e-16 \*\*\*  
## JWMNP 1.431e-03 7.106e-05 20.141 < 2e-16 \*\*\*  
## JWTR02 1.093e-02 1.481e-02 0.737 0.460822   
## JWTR03 3.399e-02 4.974e-02 0.683 0.494321   
## JWTR04 2.488e-01 1.612e-02 15.434 < 2e-16 \*\*\*  
## JWTR05 3.626e-01 1.985e-02 18.263 < 2e-16 \*\*\*  
## JWTR06 3.288e-01 4.727e-02 6.955 3.53e-12 \*\*\*  
## JWTR07 8.782e-02 2.524e-02 3.479 0.000503 \*\*\*  
## JWTR08 -2.325e-02 2.305e-02 -1.009 0.313043   
## JWTR09 -5.182e-02 1.791e-02 -2.894 0.003805 \*\*   
## JWTR10 -6.349e-02 1.350e-02 -4.705 2.54e-06 \*\*\*  
## JWTR12 9.491e-03 1.469e-02 0.646 0.518130   
## SPORDER -6.923e-02 1.313e-03 -52.723 < 2e-16 \*\*\*  
## PWGTP 4.407e-05 6.578e-06 6.700 2.09e-11 \*\*\*  
## AGEP 4.456e-03 4.849e-04 9.191 < 2e-16 \*\*\*  
## CIT3TRUE -3.097e-02 6.628e-03 -4.672 2.98e-06 \*\*\*  
## CIT4TRUE 9.808e-03 4.480e-03 2.189 0.028572 \*   
## CIT5TRUE -1.488e-02 4.987e-03 -2.983 0.002857 \*\*   
## COW2TRUE -8.112e-02 1.805e-03 -44.949 < 2e-16 \*\*\*  
## COW3TRUE -8.743e-02 1.869e-03 -46.793 < 2e-16 \*\*\*  
## COW4TRUE -1.141e-01 2.277e-03 -50.089 < 2e-16 \*\*\*  
## COW5TRUE 6.980e-02 2.941e-03 23.732 < 2e-16 \*\*\*  
## COW6TRUE -1.073e-01 2.194e-03 -48.883 < 2e-16 \*\*\*  
## COW7TRUE 9.960e-02 2.500e-03 39.844 < 2e-16 \*\*\*  
## COW8TRUE -1.725e-01 1.103e-02 -15.645 < 2e-16 \*\*\*  
## DDRS2TRUE -2.697e-02 8.954e-03 -3.012 0.002594 \*\*   
## DEYE2TRUE 2.110e-02 5.392e-03 3.913 9.13e-05 \*\*\*  
## DPHY2TRUE 2.578e-02 5.036e-03 5.119 3.07e-07 \*\*\*  
## DREM2TRUE 3.432e-02 5.451e-03 6.296 3.06e-10 \*\*\*  
## ENG2TRUE -9.786e-02 2.844e-03 -34.408 < 2e-16 \*\*\*  
## ENG3TRUE -1.272e-01 3.656e-03 -34.784 < 2e-16 \*\*\*  
## ENG4TRUE -1.138e-01 6.350e-03 -17.920 < 2e-16 \*\*\*  
## FER1TRUE 4.442e-02 5.013e-03 8.861 < 2e-16 \*\*\*  
## FER2TRUE 2.342e-02 2.036e-03 11.503 < 2e-16 \*\*\*  
## GCL2TRUE 2.570e-01 2.083e-02 12.342 < 2e-16 \*\*\*  
## GCR2TRUE 8.356e-02 4.472e-02 1.869 0.061677 .   
## HINS12TRUE -1.520e-01 1.500e-03 -101.372 < 2e-16 \*\*\*  
## HINS22TRUE -1.308e-02 1.728e-03 -7.569 3.76e-14 \*\*\*  
## HINS42TRUE 9.252e-02 2.248e-03 41.164 < 2e-16 \*\*\*  
## HINS52TRUE -2.228e-02 3.320e-03 -6.711 1.94e-11 \*\*\*  
## HINS62TRUE 5.436e-02 3.758e-03 14.465 < 2e-16 \*\*\*  
## HINS72TRUE 3.229e-03 8.499e-03 0.380 0.703987   
## LANX2TRUE 1.159e-02 2.012e-03 5.758 8.50e-09 \*\*\*  
## MAR2TRUE -2.623e-02 3.210e-03 -8.173 3.02e-16 \*\*\*  
## MAR3TRUE -4.197e-02 1.670e-03 -25.131 < 2e-16 \*\*\*  
## MAR4TRUE -4.869e-02 3.359e-03 -14.493 < 2e-16 \*\*\*  
## MARHD2TRUE -2.075e-02 4.490e-03 -4.621 3.81e-06 \*\*\*  
## MARHT2TRUE -5.534e-03 1.472e-03 -3.758 0.000171 \*\*\*  
## MARHT3TRUE -2.639e-02 2.677e-03 -9.860 < 2e-16 \*\*\*  
## MARHYP -3.978e-04 6.967e-05 -5.710 1.13e-08 \*\*\*  
## MIG2TRUE -5.720e-02 8.815e-03 -6.489 8.64e-11 \*\*\*  
## MIG3TRUE -1.420e-02 1.707e-03 -8.323 < 2e-16 \*\*\*  
## NWAB2TRUE -2.978e-02 9.740e-03 -3.057 0.002234 \*\*   
## NWAV5TRUE 7.657e-03 5.268e-03 1.454 0.146062   
## NWLA3TRUE 3.213e-03 1.069e-02 0.301 0.763636   
## NWLK3TRUE 8.442e-02 7.974e-03 10.586 < 2e-16 \*\*\*  
## NWRE2TRUE 6.009e-02 1.167e-02 5.149 2.62e-07 \*\*\*  
## RELP01TRUE -1.579e-01 1.740e-03 -90.721 < 2e-16 \*\*\*  
## RELP02TRUE -2.422e-01 4.139e-03 -58.517 < 2e-16 \*\*\*  
## RELP03TRUE -2.222e-01 2.175e-02 -10.218 < 2e-16 \*\*\*  
## RELP04TRUE -2.501e-01 1.523e-02 -16.422 < 2e-16 \*\*\*  
## RELP05TRUE -2.344e-01 7.721e-03 -30.357 < 2e-16 \*\*\*  
## RELP06TRUE -2.428e-01 7.102e-03 -34.192 < 2e-16 \*\*\*  
## RELP07TRUE -2.017e-01 1.581e-02 -12.757 < 2e-16 \*\*\*  
## RELP08TRUE -3.037e-01 1.467e-02 -20.705 < 2e-16 \*\*\*  
## RELP09TRUE -2.874e-01 7.891e-03 -36.425 < 2e-16 \*\*\*  
## RELP10TRUE -2.442e-01 8.130e-03 -30.035 < 2e-16 \*\*\*  
## RELP11TRUE -2.412e-01 1.051e-02 -22.950 < 2e-16 \*\*\*  
## RELP12TRUE -2.277e-01 7.150e-03 -31.849 < 2e-16 \*\*\*  
## RELP13TRUE -2.062e-01 4.674e-03 -44.118 < 2e-16 \*\*\*  
## RELP15TRUE -2.442e-01 7.903e-03 -30.895 < 2e-16 \*\*\*  
## RELP17TRUE -2.786e-01 1.332e-02 -20.911 < 2e-16 \*\*\*  
## SCHL04TRUE -1.043e-01 3.053e-02 -3.416 0.000635 \*\*\*  
## SCHL05TRUE -1.342e-01 2.178e-02 -6.160 7.27e-10 \*\*\*  
## SCHL06TRUE -1.007e-01 1.467e-02 -6.863 6.73e-12 \*\*\*  
## SCHL07TRUE -1.055e-01 1.708e-02 -6.175 6.61e-10 \*\*\*  
## SCHL08TRUE -1.064e-01 1.370e-02 -7.771 7.82e-15 \*\*\*  
## SCHL09TRUE -1.012e-01 6.870e-03 -14.725 < 2e-16 \*\*\*  
## SCHL10TRUE -1.151e-01 1.224e-02 -9.400 < 2e-16 \*\*\*  
## SCHL11TRUE -8.077e-02 7.093e-03 -11.386 < 2e-16 \*\*\*  
## SCHL12TRUE -1.061e-01 5.992e-03 -17.709 < 2e-16 \*\*\*  
## SCHL13TRUE -1.301e-01 5.561e-03 -23.397 < 2e-16 \*\*\*  
## SCHL14TRUE -1.185e-01 5.176e-03 -22.892 < 2e-16 \*\*\*  
## SCHL15TRUE -9.016e-02 4.420e-03 -20.397 < 2e-16 \*\*\*  
## SCHL16TRUE -6.246e-02 2.171e-03 -28.772 < 2e-16 \*\*\*  
## SCHL17TRUE -8.702e-02 3.278e-03 -26.545 < 2e-16 \*\*\*  
## SCHL19TRUE 3.397e-02 2.304e-03 14.741 < 2e-16 \*\*\*  
## SCHL20TRUE 6.445e-02 2.424e-03 26.591 < 2e-16 \*\*\*  
## SCHL22TRUE 1.233e-01 2.478e-03 49.751 < 2e-16 \*\*\*  
## SCHL23TRUE 3.437e-01 8.596e-03 39.985 < 2e-16 \*\*\*  
## SCHL24TRUE 2.175e-01 6.802e-03 31.977 < 2e-16 \*\*\*  
## SEX2TRUE -7.976e-02 2.289e-02 -3.484 0.000493 \*\*\*  
## WKHP 1.365e-02 4.765e-05 286.546 < 2e-16 \*\*\*  
## WKW2TRUE -7.342e-02 3.505e-03 -20.948 < 2e-16 \*\*\*  
## WKW3TRUE -1.582e-01 2.339e-03 -67.629 < 2e-16 \*\*\*  
## WKW4TRUE -2.789e-01 2.820e-03 -98.897 < 2e-16 \*\*\*  
## WKW5TRUE -4.105e-01 3.867e-03 -106.155 < 2e-16 \*\*\*  
## WKW6TRUE -5.210e-01 3.940e-03 -132.240 < 2e-16 \*\*\*  
## DECADE3TRUE 3.593e-02 6.193e-03 5.801 6.58e-09 \*\*\*  
## DECADE4TRUE 2.350e-02 4.386e-03 5.359 8.37e-08 \*\*\*  
## DECADE7TRUE -2.040e-02 3.331e-03 -6.125 9.10e-10 \*\*\*  
## DECADE8TRUE -5.498e-02 4.133e-03 -13.302 < 2e-16 \*\*\*  
## DIS2TRUE 4.190e-02 4.841e-03 8.655 < 2e-16 \*\*\*  
## DRIVESP1TRUE -1.330e-02 1.269e-02 -1.049 0.294402   
## DRIVESP2TRUE -6.237e-02 1.281e-02 -4.870 1.12e-06 \*\*\*  
## DRIVESP3TRUE -5.937e-02 1.334e-02 -4.451 8.55e-06 \*\*\*  
## DRIVESP4TRUE -5.789e-02 1.438e-02 -4.025 5.70e-05 \*\*\*  
## DRIVESP5TRUE -3.096e-02 1.578e-02 -1.962 0.049801 \*   
## MSP2TRUE -1.459e-02 3.143e-03 -4.642 3.45e-06 \*\*\*  
## PAOC1TRUE -9.965e-02 2.304e-02 -4.324 1.53e-05 \*\*\*  
## PAOC2TRUE -1.329e-01 2.291e-02 -5.803 6.53e-09 \*\*\*  
## PAOC4TRUE -1.321e-01 2.288e-02 -5.773 7.80e-09 \*\*\*  
## QTRBIR3TRUE 3.229e-03 1.189e-03 2.715 0.006626 \*\*   
## RACAIAN1TRUE -2.852e-02 5.081e-03 -5.612 2.00e-08 \*\*\*  
## RACASN1TRUE 6.301e-02 4.437e-03 14.203 < 2e-16 \*\*\*  
## RACBLK1TRUE -6.564e-02 4.108e-03 -15.980 < 2e-16 \*\*\*  
## RACPI1TRUE -2.557e-02 1.224e-02 -2.088 0.036787 \*   
## RACWHT1TRUE 2.257e-02 3.879e-03 5.817 6.00e-09 \*\*\*  
## SCIENGRLP1TRUE 3.276e-01 3.554e-03 92.186 < 2e-16 \*\*\*  
## SCIENGRLP2TRUE 2.506e-01 2.330e-03 107.536 < 2e-16 \*\*\*  
## WAOB2TRUE 1.641e-02 5.217e-02 0.315 0.753068   
## WAOB3TRUE -1.879e-02 4.167e-03 -4.508 6.54e-06 \*\*\*  
## WAOB5TRUE 6.072e-02 4.846e-03 12.528 < 2e-16 \*\*\*  
## WAOB6TRUE -2.510e-02 6.657e-03 -3.771 0.000163 \*\*\*  
## WAOB7TRUE 1.233e-01 8.684e-03 14.196 < 2e-16 \*\*\*  
## WAOB8TRUE 1.105e-01 1.535e-02 7.198 6.13e-13 \*\*\*  
## AGEP\_HINS31 1.562e-03 2.448e-04 6.382 1.75e-10 \*\*\*  
## SCIENGP\_SCHL01 -9.545e-02 5.379e-03 -17.745 < 2e-16 \*\*\*  
## SCIENGP1\_SCHL22 1.123e-01 3.115e-03 36.039 < 2e-16 \*\*\*  
## SCIENGP1\_SCHL23 2.355e-01 5.968e-03 39.456 < 2e-16 \*\*\*  
## SCIENGP1\_SCHL24 1.742e-01 7.863e-03 22.154 < 2e-16 \*\*\*  
## SCIENGRLP1\_SCHL22 7.894e-03 5.833e-03 1.353 0.175985   
## SCIENGRLP2\_SCHL23 1.390e-02 9.687e-03 1.435 0.151276   
## SCIENGRLP1\_SCHL24 7.851e-02 1.269e-02 6.188 6.08e-10 \*\*\*  
## AGEP\_VETERAN 5.122e-04 1.336e-04 3.834 0.000126 \*\*\*  
## AGEP\_GCL 6.289e-03 9.141e-04 6.880 5.99e-12 \*\*\*  
## DOUT2TRUE -7.836e-03 6.693e-03 -1.171 0.241662   
## MARHD8TRUE -5.994e-03 1.348e-02 -0.445 0.656588   
## NWAB3TRUE 8.029e-04 1.048e-02 0.077 0.938949   
## RACNH1TRUE -1.854e-03 1.226e-02 -0.151 0.879789   
## RACSOR1TRUE 8.821e-03 4.479e-03 1.969 0.048897 \*   
## AGEP\_GCL2 6.915e-04 4.779e-04 1.447 0.147896   
## CIT2TRUE -4.371e-02 5.260e-02 -0.831 0.406013   
## DEAR2TRUE 1.772e-03 5.003e-03 0.354 0.723115   
## GCL1TRUE 1.786e-01 4.101e-02 4.355 1.33e-05 \*\*\*  
## NWLA2TRUE -1.020e-02 1.103e-02 -0.925 0.354919   
## DECADE5TRUE 1.040e-03 3.506e-03 0.297 0.766786   
## SCIENGP1\_SCHL21 7.754e-02 2.292e-03 33.827 < 2e-16 \*\*\*  
## VETERAN -1.758e-02 7.315e-03 -2.404 0.016228 \*   
## GCM1TRUE 1.106e-02 1.711e-02 0.646 0.518256   
## GCM2TRUE -6.377e-03 1.695e-02 -0.376 0.706680   
## GCM4TRUE 2.230e-02 1.348e-02 1.654 0.098059 .   
## HINS32TRUE 4.964e-02 1.664e-02 2.983 0.002857 \*\*   
## NWAV3TRUE -2.263e-02 8.308e-03 -2.723 0.006466 \*\*   
## SCHL02TRUE -3.983e-02 3.363e-02 -1.184 0.236227   
## SCHL03TRUE -6.848e-02 3.793e-02 -1.805 0.071006 .   
## DECADE1TRUE 1.600e-02 3.138e-02 0.510 0.610265   
## DECADE2TRUE 5.866e-03 1.042e-02 0.563 0.573573   
## PAOC3TRUE -1.105e-01 2.305e-02 -4.795 1.63e-06 \*\*\*  
## AGEP\_GCR1 1.164e-03 7.613e-04 1.529 0.126288   
## GCM3TRUE 2.665e-03 1.223e-02 0.218 0.827452   
## NWAV2TRUE -8.294e-03 1.418e-02 -0.585 0.558620   
## NWLK2TRUE 7.811e-02 7.118e-03 10.973 < 2e-16 \*\*\*  
## NWRE3TRUE 3.179e-02 1.178e-02 2.698 0.006973 \*\*   
## QTRBIR2TRUE 2.164e-03 1.219e-03 1.776 0.075757 .   
## SameResidenceWorkplaceTRUE:JWMNP 4.053e-06 7.287e-05 0.056 0.955644   
## JWMNP:JWTR02 -4.712e-04 1.278e-04 -3.688 0.000226 \*\*\*  
## JWMNP:JWTR03 -4.075e-04 1.036e-03 -0.393 0.693954   
## JWMNP:JWTR04 -3.007e-03 1.743e-04 -17.252 < 2e-16 \*\*\*  
## JWMNP:JWTR05 -1.646e-03 1.913e-04 -8.605 < 2e-16 \*\*\*  
## JWMNP:JWTR06 -1.650e-03 6.101e-04 -2.704 0.006852 \*\*   
## JWMNP:JWTR07 -1.013e-03 7.045e-04 -1.439 0.150256   
## JWMNP:JWTR08 3.503e-04 6.298e-04 0.556 0.578017   
## JWMNP:JWTR09 1.478e-03 4.447e-04 3.325 0.000885 \*\*\*  
## JWMNP:JWTR10 4.182e-04 2.490e-04 1.680 0.093044 .   
## JWMNP:JWTR12 -4.323e-05 1.213e-04 -0.356 0.721470   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.435 on 781192 degrees of freedom  
## Multiple R-squared: 0.5123, Adjusted R-squared: 0.5122   
## F-statistic: 4662 on 176 and 781192 DF, p-value: < 2.2e-16

# Test Prediction  
pred.olsDLasso.1 <- predict(olsDLasso1, newdata = dattemp[-train,])  
summary(pred.olsDLasso.1)

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## -0.3127 1.9878 2.2644 2.2562 2.5408 4.2571

length(na.omit(pred.olsDLasso.1)) # count remaining observations

## [1] 195343

# test error  
mse.1 <- mean((pred.olsDLasso.1-log(dattemp[-train,]$IncomePovertyRatio))^2, na.rm=T)  
#cat("\nMSE\n")  
mse.1

## [1] 0.1894029

## Result 1 Analysis & Hypothesis Testing

# 3 Ways of getting Test R2  
y <- log(dattemp[-train,]$IncomePovertyRatio)-mean(log(dattemp[-train,]$IncomePovertyRatio))  
yhat <- pred.olsDLasso.1-mean(pred.olsDLasso.1)  
u <- y - yhat  
# 1:  
# R2 = yhat\*yhat/yTy  
r2\_1 <- (yhat %\*% yhat)/(y %\*% y)  
r2\_1

## [,1]  
## [1,] 0.5125579

# 2:  
# R2 = 1- SSR/SST = 1- uTu/yTy  
r2\_2 <- 1 - (u %\*% u)/(y %\*% y)  
r2\_2

## [,1]  
## [1,] 0.5089613

# 3:  
# R2 = corr(y, yhat)^2, "fair r-squared"  
r2\_3 <- cor.test(y, yhat, use = "complete.obs")  
# now, square the correlation coefficient  
r2\_3

##   
## Pearson's product-moment correlation  
##   
## data: y and yhat  
## t = 449.97, df = 195341, p-value < 2.2e-16  
## alternative hypothesis: true correlation is not equal to 0  
## 95 percent confidence interval:  
## 0.7112352 0.7155903  
## sample estimates:  
## cor   
## 0.7134197

r2\_3$estimate^2

## cor   
## 0.5089676

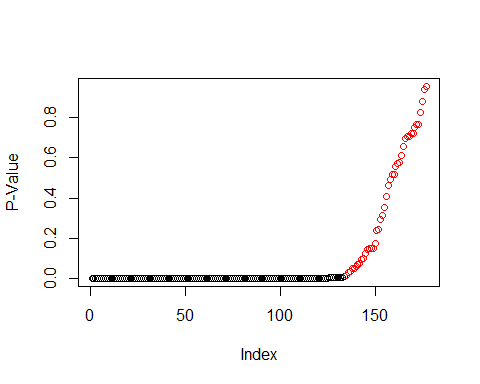
# or r2\_3 =   
require(miscTools)

## Loading required package: miscTools

rSquared(log(dattemp[-train,]$IncomePovertyRatio), resid = (log(dattemp[-train,]$IncomePovertyRatio)-pred.olsDLasso.1))

## [,1]  
## [1,] 0.5089606

# False Discovery Rate control  
p <- as.data.frame(DMLresult$coefficients[,4])  
sigcode <- cut(p[,1], breaks = c(-Inf, 0.001, 0.01, 0.05, 0.1, 1),   
 labels = c("\*\*\*", "\*\*", "\*", ".", " "))  
p$"" <- sigcode  
  
# sort by increasing p-value  
p <- p[order(p$`DMLresult$coefficients[, 4]`),]  
p$BY <- 0  
m <- nrow(p)  
Q = 0.10 # 10%  
cm=0  
for (ii in 1:m) {  
 cm = cm + 1/ii  
 p[ii,3] <- ii/m/cm\*Q  
}  
noreject <- (!(p[,1] < p[,3]))  
plot(p$`DMLresult$coefficients[, 4]`,ylab="P-Value", col = ifelse(noreject,'red','black'))



noreject <- which(noreject)  
p <- p[noreject,] # these one's we cannot reject the null  
names(p) <- c("p-value","Sig. Level","BY Stat")  
p

## p-value Sig. Level BY Stat  
## VETERAN 0.01622773 \* 0.01390240  
## CIT4TRUE 0.02857226 \* 0.01398663  
## RACPI1TRUE 0.03678683 \* 0.01407078  
## RACSOR1TRUE 0.04889671 \* 0.01415484  
## DRIVESP5TRUE 0.04980121 \* 0.01423881  
## GCR2TRUE 0.06167725 . 0.01432270  
## SCHL03TRUE 0.07100579 . 0.01440650  
## QTRBIR2TRUE 0.07575731 . 0.01449022  
## JWMNP:JWTR10 0.09304410 . 0.01457386  
## GCM4TRUE 0.09805948 . 0.01465741  
## AGEP\_GCR1 0.12628818 0.01474089  
## NWAV5TRUE 0.14606246 0.01482428  
## AGEP\_GCL2 0.14789649 0.01490759  
## JWMNP:JWTR07 0.15025585 0.01499082  
## SCIENGRLP2\_SCHL23 0.15127602 0.01507397  
## SCIENGRLP1\_SCHL22 0.17598520 0.01515704  
## SCHL02TRUE 0.23622737 0.01524004  
## DOUT2TRUE 0.24166238 0.01532296  
## DRIVESP1TRUE 0.29440172 0.01540580  
## JWTR08 0.31304290 0.01548857  
## NWLA2TRUE 0.35491891 0.01557126  
## CIT2TRUE 0.40601337 0.01565387  
## JWTR02 0.46082218 0.01573642  
## JWTR03 0.49432147 0.01581889  
## JWTR12 0.51813036 0.01590128  
## GCM1TRUE 0.51825607 0.01598361  
## NWAV2TRUE 0.55862008 0.01606586  
## DECADE2TRUE 0.57357341 0.01614804  
## JWMNP:JWTR08 0.57801715 0.01623016  
## DECADE1TRUE 0.61026469 0.01631220  
## MARHD8TRUE 0.65658784 0.01639417  
## JWMNP:JWTR03 0.69395359 0.01647607  
## HINS72TRUE 0.70398697 0.01655791  
## GCM2TRUE 0.70667974 0.01663968  
## JWMNP:JWTR12 0.72146983 0.01672138  
## DEAR2TRUE 0.72311468 0.01680301  
## WAOB2TRUE 0.75306776 0.01688458  
## NWLA3TRUE 0.76363569 0.01696608  
## DECADE5TRUE 0.76678592 0.01704751  
## GCM3TRUE 0.82745213 0.01712888  
## RACNH1TRUE 0.87978946 0.01721019  
## NWAB3TRUE 0.93894933 0.01729143  
## SameResidenceWorkplaceTRUE:JWMNP 0.95564425 0.01737261

# get BY-adjusted p-values  
pBY <- as.data.frame(p.adjust(p[,1], method = "BY")) #Benjamini-Yekutieli  
rownames(pBY) <- rownames(p)  
adjsigcode <- cut(pBY[,1], breaks = c(-Inf, 0.001, 0.01, 0.05, 0.1, 1),   
 labels = c("\*\*\*", "\*\*", "\*", ".", " "))  
pBY$"" <- adjsigcode  
  
# compare p-values for non-rejected  
fdr <- cbind.data.frame(p[,c(1,2)], pBY)  
#rownames(fdr) <- getName  
colnames(fdr) <- c("Original","Sig. Level", "FDR Adj.","Sig. Level")  
fdr

## Original Sig. Level FDR Adj. Sig. Level  
## VETERAN 0.01622773 \* 1   
## CIT4TRUE 0.02857226 \* 1   
## RACPI1TRUE 0.03678683 \* 1   
## RACSOR1TRUE 0.04889671 \* 1   
## DRIVESP5TRUE 0.04980121 \* 1   
## GCR2TRUE 0.06167725 . 1   
## SCHL03TRUE 0.07100579 . 1   
## QTRBIR2TRUE 0.07575731 . 1   
## JWMNP:JWTR10 0.09304410 . 1   
## GCM4TRUE 0.09805948 . 1   
## AGEP\_GCR1 0.12628818 1   
## NWAV5TRUE 0.14606246 1   
## AGEP\_GCL2 0.14789649 1   
## JWMNP:JWTR07 0.15025585 1   
## SCIENGRLP2\_SCHL23 0.15127602 1   
## SCIENGRLP1\_SCHL22 0.17598520 1   
## SCHL02TRUE 0.23622737 1   
## DOUT2TRUE 0.24166238 1   
## DRIVESP1TRUE 0.29440172 1   
## JWTR08 0.31304290 1   
## NWLA2TRUE 0.35491891 1   
## CIT2TRUE 0.40601337 1   
## JWTR02 0.46082218 1   
## JWTR03 0.49432147 1   
## JWTR12 0.51813036 1   
## GCM1TRUE 0.51825607 1   
## NWAV2TRUE 0.55862008 1   
## DECADE2TRUE 0.57357341 1   
## JWMNP:JWTR08 0.57801715 1   
## DECADE1TRUE 0.61026469 1   
## MARHD8TRUE 0.65658784 1   
## JWMNP:JWTR03 0.69395359 1   
## HINS72TRUE 0.70398697 1   
## GCM2TRUE 0.70667974 1   
## JWMNP:JWTR12 0.72146983 1   
## DEAR2TRUE 0.72311468 1   
## WAOB2TRUE 0.75306776 1   
## NWLA3TRUE 0.76363569 1   
## DECADE5TRUE 0.76678592 1   
## GCM3TRUE 0.82745213 1   
## RACNH1TRUE 0.87978946 1   
## NWAB3TRUE 0.93894933 1   
## SameResidenceWorkplaceTRUE:JWMNP 0.95564425 1

# BP test for heteroskedasticity  
bpres1 <- bptest(olsDLasso1, data = dattemp[-train,]) #reject homoskedasticity if p-value is small  
bpres1

##   
## studentized Breusch-Pagan test  
##   
## data: olsDLasso1  
## BP = 58372, df = 176, p-value < 2.2e-16

# F-test  
null = c("SameResidenceWorkplaceTRUE","JWMNP",  
 "JWTR02","JWTR03","JWTR04","JWTR05","JWTR06","JWTR07","JWTR08",  
 "JWTR09","JWTR10","JWTR12")  
if (bpres1$p.value >= 0.001) { # homoskedastic  
 linearHypothesis(olsDLasso1, null, vcov = hccm(olsDLasso1, type = "hc0")) # classical White VCOV  
} else {  
 linearHypothesis(olsDLasso1, null) # default homoskedastic error  
}

## Linear hypothesis test  
##   
## Hypothesis:  
## SameResidenceWorkplaceTRUE = 0  
## JWMNP = 0  
## JWTR02 = 0  
## JWTR03 = 0  
## JWTR04 = 0  
## JWTR05 = 0  
## JWTR06 = 0  
## JWTR07 = 0  
## JWTR08 = 0  
## JWTR09 = 0  
## JWTR10 = 0  
## JWTR12 = 0  
##   
## Model 1: restricted model  
## Model 2: log(IncomePovertyRatio) ~ SameResidenceWorkplace \* JWMNP + JWTR \*   
## JWMNP + SPORDER + PWGTP + AGEP + CIT3 + CIT4 + CIT5 + COW2 +   
## COW3 + COW4 + COW5 + COW6 + COW7 + COW8 + DDRS2 + DEYE2 +   
## DPHY2 + DREM2 + ENG2 + ENG3 + ENG4 + FER1 + FER2 + GCL2 +   
## GCR2 + HINS12 + HINS22 + HINS42 + HINS52 + HINS62 + HINS72 +   
## LANX2 + MAR2 + MAR3 + MAR4 + MARHD2 + MARHT2 + MARHT3 + MARHYP +   
## MIG2 + MIG3 + NWAB2 + NWAV5 + NWLA3 + NWLK3 + NWRE2 + RELP01 +   
## RELP02 + RELP03 + RELP04 + RELP05 + RELP06 + RELP07 + RELP08 +   
## RELP09 + RELP10 + RELP11 + RELP12 + RELP13 + RELP15 + RELP17 +   
## SCHL04 + SCHL05 + SCHL06 + SCHL07 + SCHL08 + SCHL09 + SCHL10 +   
## SCHL11 + SCHL12 + SCHL13 + SCHL14 + SCHL15 + SCHL16 + SCHL17 +   
## SCHL19 + SCHL20 + SCHL22 + SCHL23 + SCHL24 + SEX2 + WKHP +   
## WKW2 + WKW3 + WKW4 + WKW5 + WKW6 + DECADE3 + DECADE4 + DECADE7 +   
## DECADE8 + DIS2 + DRIVESP1 + DRIVESP2 + DRIVESP3 + DRIVESP4 +   
## DRIVESP5 + MSP2 + PAOC1 + PAOC2 + PAOC4 + QTRBIR3 + RACAIAN1 +   
## RACASN1 + RACBLK1 + RACPI1 + RACWHT1 + SCIENGRLP1 + SCIENGRLP2 +   
## WAOB2 + WAOB3 + WAOB5 + WAOB6 + WAOB7 + WAOB8 + AGEP\_HINS31 +   
## SCIENGP\_SCHL01 + SCIENGP1\_SCHL22 + SCIENGP1\_SCHL23 + SCIENGP1\_SCHL24 +   
## SCIENGRLP1\_SCHL22 + SCIENGRLP2\_SCHL23 + SCIENGRLP1\_SCHL24 +   
## AGEP\_VETERAN + AGEP\_GCL + DOUT2 + MARHD8 + NWAB3 + RACNH1 +   
## RACSOR1 + AGEP\_GCL2 + CIT2 + DEAR2 + GCL1 + NWLA2 + DECADE5 +   
## SCIENGP1\_SCHL21 + VETERAN + GCM1 + GCM2 + GCM4 + HINS32 +   
## NWAV3 + SCHL02 + SCHL03 + DECADE1 + DECADE2 + PAOC3 + AGEP\_GCR1 +   
## GCM3 + NWAV2 + NWLK2 + NWRE3 + QTRBIR2  
##   
## Res.Df RSS Df Sum of Sq F Pr(>F)   
## 1 781204 148776   
## 2 781192 147823 12 953.35 419.85 < 2.2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

null = c("JWMNP",  
 "JWTR02","JWTR03","JWTR04","JWTR05","JWTR06","JWTR07","JWTR08",  
 "JWTR09","JWTR10","JWTR12",  
 "JWMNP:JWTR02","JWMNP:JWTR03","JWMNP:JWTR04","JWMNP:JWTR05","JWMNP:JWTR06",  
 "JWMNP:JWTR07","JWMNP:JWTR08","JWMNP:JWTR09","JWMNP:JWTR10","JWMNP:JWTR12")  
if (bpres1$p.value >= 0.001) { # homoskedastic  
 linearHypothesis(olsDLasso1, null, vcov = hccm(olsDLasso1, type = "hc0")) # classical White VCOV  
} else {  
 linearHypothesis(olsDLasso1, null)  
}

## Linear hypothesis test  
##   
## Hypothesis:  
## JWMNP = 0  
## JWTR02 = 0  
## JWTR03 = 0  
## JWTR04 = 0  
## JWTR05 = 0  
## JWTR06 = 0  
## JWTR07 = 0  
## JWTR08 = 0  
## JWTR09 = 0  
## JWTR10 = 0  
## JWTR12 = 0  
## JWMNP:JWTR02 = 0  
## JWMNP:JWTR03 = 0  
## JWMNP:JWTR04 = 0  
## JWMNP:JWTR05 = 0  
## JWMNP:JWTR06 = 0  
## JWMNP:JWTR07 = 0  
## JWMNP:JWTR08 = 0  
## JWMNP:JWTR09 = 0  
## JWMNP:JWTR10 = 0  
## JWMNP:JWTR12 = 0  
##   
## Model 1: restricted model  
## Model 2: log(IncomePovertyRatio) ~ SameResidenceWorkplace \* JWMNP + JWTR \*   
## JWMNP + SPORDER + PWGTP + AGEP + CIT3 + CIT4 + CIT5 + COW2 +   
## COW3 + COW4 + COW5 + COW6 + COW7 + COW8 + DDRS2 + DEYE2 +   
## DPHY2 + DREM2 + ENG2 + ENG3 + ENG4 + FER1 + FER2 + GCL2 +   
## GCR2 + HINS12 + HINS22 + HINS42 + HINS52 + HINS62 + HINS72 +   
## LANX2 + MAR2 + MAR3 + MAR4 + MARHD2 + MARHT2 + MARHT3 + MARHYP +   
## MIG2 + MIG3 + NWAB2 + NWAV5 + NWLA3 + NWLK3 + NWRE2 + RELP01 +   
## RELP02 + RELP03 + RELP04 + RELP05 + RELP06 + RELP07 + RELP08 +   
## RELP09 + RELP10 + RELP11 + RELP12 + RELP13 + RELP15 + RELP17 +   
## SCHL04 + SCHL05 + SCHL06 + SCHL07 + SCHL08 + SCHL09 + SCHL10 +   
## SCHL11 + SCHL12 + SCHL13 + SCHL14 + SCHL15 + SCHL16 + SCHL17 +   
## SCHL19 + SCHL20 + SCHL22 + SCHL23 + SCHL24 + SEX2 + WKHP +   
## WKW2 + WKW3 + WKW4 + WKW5 + WKW6 + DECADE3 + DECADE4 + DECADE7 +   
## DECADE8 + DIS2 + DRIVESP1 + DRIVESP2 + DRIVESP3 + DRIVESP4 +   
## DRIVESP5 + MSP2 + PAOC1 + PAOC2 + PAOC4 + QTRBIR3 + RACAIAN1 +   
## RACASN1 + RACBLK1 + RACPI1 + RACWHT1 + SCIENGRLP1 + SCIENGRLP2 +   
## WAOB2 + WAOB3 + WAOB5 + WAOB6 + WAOB7 + WAOB8 + AGEP\_HINS31 +   
## SCIENGP\_SCHL01 + SCIENGP1\_SCHL22 + SCIENGP1\_SCHL23 + SCIENGP1\_SCHL24 +   
## SCIENGRLP1\_SCHL22 + SCIENGRLP2\_SCHL23 + SCIENGRLP1\_SCHL24 +   
## AGEP\_VETERAN + AGEP\_GCL + DOUT2 + MARHD8 + NWAB3 + RACNH1 +   
## RACSOR1 + AGEP\_GCL2 + CIT2 + DEAR2 + GCL1 + NWLA2 + DECADE5 +   
## SCIENGP1\_SCHL21 + VETERAN + GCM1 + GCM2 + GCM4 + HINS32 +   
## NWAV3 + SCHL02 + SCHL03 + DECADE1 + DECADE2 + PAOC3 + AGEP\_GCR1 +   
## GCM3 + NWAV2 + NWLK2 + NWRE3 + QTRBIR2  
##   
## Res.Df RSS Df Sum of Sq F Pr(>F)   
## 1 781213 148435   
## 2 781192 147823 21 612.12 154.04 < 2.2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

null = c("SameResidenceWorkplaceTRUE","JWMNP",  
 "JWTR02","JWTR03","JWTR04","JWTR05","JWTR06","JWTR07","JWTR08",  
 "JWTR09","JWTR10","JWTR12",  
 "SameResidenceWorkplaceTRUE:JWMNP",  
 "JWMNP:JWTR02","JWMNP:JWTR03","JWMNP:JWTR04","JWMNP:JWTR05","JWMNP:JWTR06",  
 "JWMNP:JWTR07","JWMNP:JWTR08","JWMNP:JWTR09","JWMNP:JWTR10","JWMNP:JWTR12")  
if (bpres1$p.value >= 0.001) { # homoskedastic  
 linearHypothesis(olsDLasso1, null, vcov = hccm(olsDLasso1, type = "hc0")) # classical White VCOV  
} else {  
 linearHypothesis(olsDLasso1, null)  
}

## Linear hypothesis test  
##   
## Hypothesis:  
## SameResidenceWorkplaceTRUE = 0  
## JWMNP = 0  
## JWTR02 = 0  
## JWTR03 = 0  
## JWTR04 = 0  
## JWTR05 = 0  
## JWTR06 = 0  
## JWTR07 = 0  
## JWTR08 = 0  
## JWTR09 = 0  
## JWTR10 = 0  
## JWTR12 = 0  
## SameResidenceWorkplaceTRUE:JWMNP = 0  
## JWMNP:JWTR02 = 0  
## JWMNP:JWTR03 = 0  
## JWMNP:JWTR04 = 0  
## JWMNP:JWTR05 = 0  
## JWMNP:JWTR06 = 0  
## JWMNP:JWTR07 = 0  
## JWMNP:JWTR08 = 0  
## JWMNP:JWTR09 = 0  
## JWMNP:JWTR10 = 0  
## JWMNP:JWTR12 = 0  
##   
## Model 1: restricted model  
## Model 2: log(IncomePovertyRatio) ~ SameResidenceWorkplace \* JWMNP + JWTR \*   
## JWMNP + SPORDER + PWGTP + AGEP + CIT3 + CIT4 + CIT5 + COW2 +   
## COW3 + COW4 + COW5 + COW6 + COW7 + COW8 + DDRS2 + DEYE2 +   
## DPHY2 + DREM2 + ENG2 + ENG3 + ENG4 + FER1 + FER2 + GCL2 +   
## GCR2 + HINS12 + HINS22 + HINS42 + HINS52 + HINS62 + HINS72 +   
## LANX2 + MAR2 + MAR3 + MAR4 + MARHD2 + MARHT2 + MARHT3 + MARHYP +   
## MIG2 + MIG3 + NWAB2 + NWAV5 + NWLA3 + NWLK3 + NWRE2 + RELP01 +   
## RELP02 + RELP03 + RELP04 + RELP05 + RELP06 + RELP07 + RELP08 +   
## RELP09 + RELP10 + RELP11 + RELP12 + RELP13 + RELP15 + RELP17 +   
## SCHL04 + SCHL05 + SCHL06 + SCHL07 + SCHL08 + SCHL09 + SCHL10 +   
## SCHL11 + SCHL12 + SCHL13 + SCHL14 + SCHL15 + SCHL16 + SCHL17 +   
## SCHL19 + SCHL20 + SCHL22 + SCHL23 + SCHL24 + SEX2 + WKHP +   
## WKW2 + WKW3 + WKW4 + WKW5 + WKW6 + DECADE3 + DECADE4 + DECADE7 +   
## DECADE8 + DIS2 + DRIVESP1 + DRIVESP2 + DRIVESP3 + DRIVESP4 +   
## DRIVESP5 + MSP2 + PAOC1 + PAOC2 + PAOC4 + QTRBIR3 + RACAIAN1 +   
## RACASN1 + RACBLK1 + RACPI1 + RACWHT1 + SCIENGRLP1 + SCIENGRLP2 +   
## WAOB2 + WAOB3 + WAOB5 + WAOB6 + WAOB7 + WAOB8 + AGEP\_HINS31 +   
## SCIENGP\_SCHL01 + SCIENGP1\_SCHL22 + SCIENGP1\_SCHL23 + SCIENGP1\_SCHL24 +   
## SCIENGRLP1\_SCHL22 + SCIENGRLP2\_SCHL23 + SCIENGRLP1\_SCHL24 +   
## AGEP\_VETERAN + AGEP\_GCL + DOUT2 + MARHD8 + NWAB3 + RACNH1 +   
## RACSOR1 + AGEP\_GCL2 + CIT2 + DEAR2 + GCL1 + NWLA2 + DECADE5 +   
## SCIENGP1\_SCHL21 + VETERAN + GCM1 + GCM2 + GCM4 + HINS32 +   
## NWAV3 + SCHL02 + SCHL03 + DECADE1 + DECADE2 + PAOC3 + AGEP\_GCR1 +   
## GCM3 + NWAV2 + NWLK2 + NWRE3 + QTRBIR2  
##   
## Res.Df RSS Df Sum of Sq F Pr(>F)   
## 1 781215 149713   
## 2 781192 147823 23 1890.1 434.27 < 2.2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

## Without Potential Endogeneity

## Model 2 Post-Double ML OLs with exogenous variable selection ----------------------------------------------------------------  
# manually delete potentially endogenous variables  
# Can X affect or cause Income or Poverty of Both?  
endog <- c("SPORDER", # household size  
 "CIT3","CIT4","CIT5", # citizenship status  
 "COW2","COW3","COW4","COW5","COW6","COW7","COW8", # class of worker  
 "DDRS2","DEYE2","DPHY2","DREM2", # disability  
 "ENG2","ENG3","ENG4","FER1","FER2", # level of english and child birth  
 "GCL2","GCR2", # grandparents with grandchildren  
 "HINS12","HINS22","HINS42","HINS52","HINS62","HINS72", # insurance  
 "MAR2","MAR3","MAR4","MARHD2","MARHT2","MARHT3", # marriage  
 "MIG2","MIG3", # migration  
 "NWAB2","NWAV5","NWLA3","NWLK3","NWRE2", # current work status  
 "RELP01","RELP02","RELP03","RELP04","RELP05","RELP06","RELP07", # relationship in household  
 "RELP08","RELP09","RELP10","RELP11","RELP12","RELP13","RELP15","RELP17"  
 )  
endog2 <- c(61:79,80,81:86, # degree, sex, work  
 91,92:96,97, # disability, num cars per ppl, marriage status,   
 102:106,107:108, # race, stem degree  
 116:122,123:124,125, # stem\*degree, age\*stuff, disability  
 126:134, # marriage, work, race, age\*stuff, citizenship, disability, work  
 136:140, # school, veteran, grandparents with grandchild  
 142:144, # insurance, work, school  
 148:152 # age\*stuff, grandparents with grandchild, work  
 )  
  
union <- union[-endog2]  
union <- union[-which(union %in% endog)] # delete them from formula  
  
# rewrite formula for OLS  
exogunionf <- paste(union, collapse = "+")  
exogformula <- paste(c("log(IncomePovertyRatio)", exogunionf), collapse = "~")  
  
# Training OLS regression post LASSO  
olsDLasso2 <- lm(exogformula, data = dattemp[train,])  
DMLresult2 <- summary(olsDLasso2)  
# Post-Double LASSO OLS only on Exogeneous vars Result  
DMLresult2

##   
## Call:  
## lm(formula = exogformula, data = dattemp[train, ])  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -2.34876 -0.39110 -0.04425 0.33410 3.11596   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) -1.428e+00 1.631e-01 -8.758 < 2e-16 \*\*\*  
## PWGTP -1.671e-04 8.701e-06 -19.200 < 2e-16 \*\*\*  
## AGEP 8.362e-03 9.287e-05 90.045 < 2e-16 \*\*\*  
## LANX2TRUE 5.025e-02 2.234e-03 22.496 < 2e-16 \*\*\*  
## MARHYP 1.625e-03 7.909e-05 20.547 < 2e-16 \*\*\*  
## DECADE3TRUE 1.416e-01 7.867e-03 18.003 < 2e-16 \*\*\*  
## DECADE4TRUE 1.505e-01 5.346e-03 28.160 < 2e-16 \*\*\*  
## DECADE7TRUE 2.341e-02 3.879e-03 6.034 1.6e-09 \*\*\*  
## DECADE8TRUE -5.670e-02 4.600e-03 -12.326 < 2e-16 \*\*\*  
## PAOC1TRUE -1.810e-01 3.869e-03 -46.789 < 2e-16 \*\*\*  
## PAOC2TRUE -2.695e-01 2.217e-03 -121.533 < 2e-16 \*\*\*  
## PAOC4TRUE -3.561e-01 1.553e-03 -229.339 < 2e-16 \*\*\*  
## QTRBIR3TRUE 4.710e-03 1.610e-03 2.926 0.00344 \*\*   
## WAOB2TRUE -1.658e-01 9.589e-03 -17.294 < 2e-16 \*\*\*  
## WAOB3TRUE -3.119e-01 3.194e-03 -97.644 < 2e-16 \*\*\*  
## WAOB5TRUE 8.521e-02 4.784e-03 17.810 < 2e-16 \*\*\*  
## WAOB6TRUE -1.018e-01 7.735e-03 -13.166 < 2e-16 \*\*\*  
## WAOB7TRUE 2.027e-01 1.084e-02 18.696 < 2e-16 \*\*\*  
## WAOB8TRUE 4.730e-02 1.933e-02 2.448 0.01439 \*   
## AGEP\_HINS31 -2.175e-04 3.219e-04 -0.676 0.49922   
## DECADE5TRUE 9.779e-02 4.130e-03 23.680 < 2e-16 \*\*\*  
## HINS32TRUE 1.916e-01 2.207e-02 8.682 < 2e-16 \*\*\*  
## DECADE1TRUE 7.548e-02 4.239e-02 1.781 0.07499 .   
## DECADE2TRUE 7.733e-02 1.371e-02 5.640 1.7e-08 \*\*\*  
## PAOC3TRUE -2.705e-01 4.101e-03 -65.946 < 2e-16 \*\*\*  
## QTRBIR2TRUE 5.347e-03 1.649e-03 3.241 0.00119 \*\*   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.5889 on 781343 degrees of freedom  
## Multiple R-squared: 0.106, Adjusted R-squared: 0.106   
## F-statistic: 3708 on 25 and 781343 DF, p-value: < 2.2e-16

# Test Prediction  
pred.olsDLasso.2 <- predict(olsDLasso2, newdata = dattemp[-train,])  
summary(pred.olsDLasso.2)

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 1.402 2.113 2.234 2.258 2.431 2.950

length(na.omit(pred.olsDLasso.2)) # count remaining observations

## [1] 195343

# test error  
mse.2 <- mean((pred.olsDLasso.2-log(dattemp[-train,]$IncomePovertyRatio))^2, na.rm=T)  
# MSE  
mse.2

## [1] 0.34607

## Result 2 Analysis & Hypothesis Testing

# 3 Ways of getting Test R2  
y2 <- log(dattemp[-train,]$IncomePovertyRatio)-mean(log(dattemp[-train,]$IncomePovertyRatio))  
yhat2 <- pred.olsDLasso.2-mean(pred.olsDLasso.2)  
u2 <- y2 - yhat2  
# 1:  
# R2 = yhat\*y/yTy  
r2\_1\_2 <- (yhat2 %\*% yhat2)/(y2 %\*% y2)  
r2\_1\_2

## [,1]  
## [1,] 0.1064778

# 2:  
# R2 = 1- SSR/SST = 1- uTu/yTy  
r2\_2\_2 <- 1 - (u2 %\*% u2)/(y2 %\*% y2)  
r2\_2\_2

## [,1]  
## [1,] 0.1028011

# 3:  
# R2 = corr(y, yhat)^2, "fair r-squared"  
r2\_3\_2 <- cor.test(y2, yhat2, use = "complete.obs")  
# now, square the correlation coefficient  
r2\_3\_2

##   
## Pearson's product-moment correlation  
##   
## data: y2 and yhat2  
## t = 149.63, df = 195341, p-value < 2.2e-16  
## alternative hypothesis: true correlation is not equal to 0  
## 95 percent confidence interval:  
## 0.3166914 0.3246485  
## sample estimates:  
## cor   
## 0.3206756

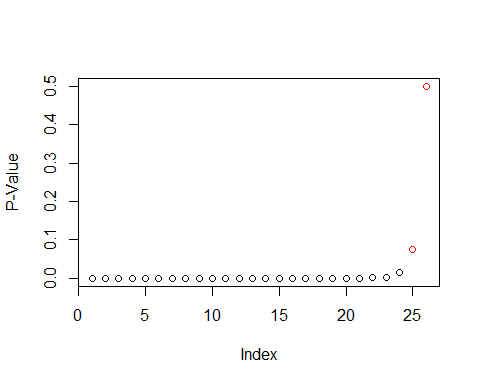
r2\_3\_2$estimate^2

## cor   
## 0.1028329

# BP test for heteroskedasticity  
bpres2 <- bptest(olsDLasso2, data = dattemp[-train,]) #reject homoskedasticity if p-value is small  
bpres2

##   
## studentized Breusch-Pagan test  
##   
## data: olsDLasso2  
## BP = 13044, df = 25, p-value < 2.2e-16

# False Discovery Rate control  
p2 <- as.data.frame(DMLresult2$coefficients[,4])  
sigcode2 <- cut(p2[,1], breaks = c(-Inf, 0.001, 0.01, 0.05, 0.1, 1),   
 labels = c("\*\*\*", "\*\*", "\*", ".", " "))  
p2$"" <- sigcode2  
  
# sort by increasing p-value  
p2 <- p2[order(p2$`DMLresult2$coefficients[, 4]`),]  
p2$BY <- 0  
m2 <- nrow(p2)  
Q = 0.10 # 10%  
cm=0  
for (ii in 1:m2) {  
 cm = cm + 1/ii  
 p2[ii,3] <- ii/m2/cm\*Q  
}  
noreject2 <- (!(p2[,1] < p2[,3]))  
plot(p2$`DMLresult2$coefficients[, 4]`,ylab="P-Value", col = ifelse(noreject2,'red','black'))



noreject2 <- which(noreject2)  
p2 <- p2[noreject2,] # these one's we cannot reject the null  
names(p2) <- c("p-value","Sig. Level","BY Stat")  
p2

## p-value Sig. Level BY Stat  
## DECADE1TRUE 0.07498765 . 0.02519782  
## AGEP\_HINS31 0.49921838 0.02594424

# get BY-adjusted p-values  
pBY2 <- as.data.frame(p.adjust(p2[,1], method = "BY")) #Benjamini-Yekutieli  
rownames(pBY2) <- rownames(p2)  
adjsigcode <- cut(pBY2[,1], breaks = c(-Inf, 0.001, 0.01, 0.05, 0.1, 1),   
 labels = c("\*\*\*", "\*\*", "\*", ".", " "))  
pBY2$"" <- adjsigcode  
  
# compare p-values for non-rejected  
fdr2 <- cbind.data.frame(p2[,c(1,2)], pBY2)  
colnames(fdr2) <- c("Original","Sig. Level", "FDR Adj.","Sig. Level")  
fdr2

## Original Sig. Level FDR Adj. Sig. Level  
## DECADE1TRUE 0.07498765 . 0.2249630   
## AGEP\_HINS31 0.49921838 0.7488276